CLAIM AMENDMENTS

Please amend the claims by canceling claims 3-4, 21-29, and 33, amending claims 1, 14, 30, and 31, and adding new claims 36-37, all without prejudice, as indicated on the following listing of all the claims in the present application after this Amendment:

1. (Currently Amended) A system for starting and controlling operation of an intelligent device comprising:

~~an~~ NAND flash memory application and file storage device configured to read and write data files, one or more of the data files including the basic input/output system (BIOS) interface;

a random access memory (RAM);

a loading logic circuit that configures the application and file storage device to operate and,

places the chip enable signal of the application and file storage device in an active state;

sends a command to the application and file storage device over a data bus of the device;

awaits the application and file storage device to change state from a busy state, while processing the command, to a ready state;

sets the command latch enable and read address latch enable signal to an inactive state;

sends a read enable signal to the application and file storage device; and

sends a write signal to the RAM,

and thereby copies a portion of the BIOS from the storage device into the RAM.

2. (Original) The system of claim 1 wherein the loading logic circuit is configured to copy the portion of the BIOS from the application and file storage device into the RAM without using a microprocessor.

3. (Canceled)

4. (Canceled)

5. (Original) The system of claim 1 wherein the loading logic circuit is contained in a field programmable gate array(FPGA).

6. (Original) The system of claim 1 wherein the loading logic circuit is contained in a programmable logic device.

7. (Original) The system of claim 1 wherein the circuit comprises board level components.

8. (Original) The system of claim 1 wherein the loading logic circuitry stores the BIOS at any location of the storage device.

9. (Original) The system of claim 1 wherein the loading logic circuitry copies the BIOS to any location in the RAM.

10. (Original) The system of claim 1 wherein the loading logic circuitry comprises a write protect mechanism that prevents the location of the storage device having the BIOS from being overwritten.

11. (Original) The system of claim 10 wherein the write protect mechanism generates a first and second write strobe signal for each write strobe signal of a microprocessor.

12. (Original) The system of claim 11 wherein the first write strobe signal is separated from the second write strobe SIGNAL by a period of time, and wherein the first write strobe signal records the location of the BIOS in the storage device and the command code to the storage device, and the second write strobe signal enables writing of the storage device.

13. (Original) The system of claim 12 wherein when the microprocessor attempts to write to the location of the BIOS in the storage device, the second write strobe signal is not generated whereby the BIOS is not overwritten.

14. (Currently Amended) A method of starting a smart device comprising:

resetting operation of a microprocessor; and thereafter

suspending operation of the microprocessor; and thereafter

enabling operation of an application and file storage device with system level circuitry not provided as part of the application and file storage device; and thereafter

copying a portion of a BIOS from ~~an~~ the application and file storage device into RAM with said system level circuitry; and thereafter

starting operation of the microprocessor.

15. (Original) The method of claim 14 wherein the storage device includes multiple BIOSs and wherein a user can select which BIOS to copy from the application and file storage device into RAM.

16. (Original) The method of claim 14 further comprising the step of reading the portion of the BIOS from the RAM with the central processing unit after the step of starting operation of the microprocessor.

17. (Original) The method of claim 14 wherein the step of copying the BIOS from a memory storage device into RAM is controlled by a state machine.

18. (Original) The method of claim 17 wherein the state machine is implemented in an ASIC.

19. (Original) The method of claim 17 wherein the state machine is implemented in an FPGA.

20. (Original) The method of claim 14 wherein the application and file storage device is a flash memory device.

21-29. (Canceled)

30. (Currently Amended) A method of providing an interface between an operating system and hardware devices comprising:

storing the interface in an application and file storage device; and thereafter

copying the interface from the application and application and file storage device into RAM without using a microprocessor by:

enabling the application and file storage device and the RAM; and thereafter

enabling an address counter to output a value; and thereafter

correlating the value with a RAM address; and thereafter

sending data from the application and file storage device over a data bus to the RAM address; and thereafter

~~The method of claim 22 wherein the step of copying the interface further comprises the usage of~~ verifying the sent data with error correction code, and

incrementing the address counter.

31. (Currently Amended) A system for booting a microprocessor controlled device comprising:

an application and file storage device having a plurality of files, said application and file storage device not incorporating a controller for operating the device;

a random access memory;

a microprocessor;

human interface devices; and

an interface for communicating between the microprocessor, the application and file storage device and the human interface devices, the interface residing in a file of the file storage

device; and means for copying a portion of the interface into the random access memory ~~without using the microprocessor~~, said means functioning to directly control the read and write operations of the application and file storage device.

32. (Original) The system of claim 31 wherein the application and file storage device comprises a non-volatile solid state memory device.

33. (Canceled)

34. (Original) The system of claim 31 further comprising a means for protecting the file on the file storage device from being overwritten.

35. (Original) The system of claim 1 wherein the loading logic circuit further comprises a means for protecting the basic input/output system (BIOS) interface on the application and file storage device from being overwritten.

36. (New) A method of operating a smart device comprising:

providing a general purpose application and file storage device for the storage of user files;

allocating a small portion of the general purpose application and file storage device for a BIOS of the device;

resetting operation of a microprocessor; and thereafter

suspending operation of the microprocessor; and thereafter

copying a portion of a BIOS from the application and file storage device into RAM by

enabling the general purpose application and file storage device and the RAM; and thereafter

enabling an address counter to output a value; and thereafter

correlating the value with a RAM address; and thereafter

sending data from the application and file storage device over a data bus to the RAM address; and thereafter

incrementing the address counter; and thereafter

starting operation of the microprocessor; and

using the general purpose application and file storage device to read and write user files, and during such usage, preventing the BIOS from being overwritten.

37. (New) A method of starting and operating a processor controlled system comprising an operating system, hardware devices, and an interface between the operating system and the hardware devices, the method comprising:

storing the interface in a directly accessed NAND flash application and file storage memory; and thereafter

copying the interface from the application and file storage memory into RAM by:

enabling the application and file storage memory and the RAM; and thereafter

enabling an address counter to output a value; and thereafter

correlating the value with a RAM address; and thereafter

sending data from the application and file storage memory; and thereafter

incrementing the address counter.